

IN THE CLAIMS:

Please cancel claims 4, 5, 15, and 16 without prejudice or disclaimer as to the subject matter contained therein.

Please amend the claims as shown in the following claims listing.

1. (Currently amended) A processor comprising:
an execution unit configured to execute one or more threads; and
a detection unit coupled to the execution unit and configured to detect whether a given thread includes [[an]] a specific identifier;
wherein said execution unit is further configured to selectively continue execution of said given thread depending upon whether said detection unit detects said specific identifier;
wherein said execution unit is configured to suspend execution of said given thread and to execute a different thread in response to receiving a global execution parameter; and
wherein in response to said detection unit detecting said specific identifier, said execution unit is configured to override said global execution parameter and to continue execution of said given thread.
2. (Currently amended) The processor of claim 1, wherein in response to said detection unit detecting [[said]] a different identifier, said execution unit is configured to suspend execution of said given thread and to execute a different thread.
3. (Currently amended) The processor of claim 2, wherein in response to said detection unit detecting that said given thread does not include said different identifier, said executing unit is configured to continue execution of said given thread.
- 4-5. (Cancelled)

6. (Currently amended) The processor of claim 1 further comprising a priority designation unit coupled to said detection unit and configured to assign a priority level to said given thread depending upon an execution environment in response to said detection unit detecting said specific identifier.
7. (Original) The processor of claim 6, wherein in response to said given thread having a priority level lower than said different thread, said execution unit is configured to suspend execution of said given thread and to execute said different thread with a higher priority level
8. (Original) The processor of claim 7, wherein in response to said given thread having a priority level the same as or higher than said different thread, said execution unit is configured to continue execution of said given thread.
9. (Currently amended) The processor of claim 1, wherein said specific identifier is a unique instruction.
10. (Currently amended) The processor of claim 1, wherein said specific identifier is a flag including one or more unused bits of any instruction of said given thread.
11. (Currently amended) The processor of claim 1, wherein said specific identifier is information associated with any instruction of said given thread.
12. (Currently amended) A method comprising:
executing one or more threads;
detecting whether a given thread includes [[an]] a specific identifier; [[and]]
selectively continuing execution of said given thread depending upon whether
said specific identifier is detected;
suspending execution of said given thread and executing a different thread in
response to receiving a global execution parameter; and

overriding said global execution parameter and continuing execution of said given thread in response to detecting said specific identifier.

13. (Currently amended) The method of claim 12, wherein in response to detecting [[said]] a different identifier, suspending execution of said given thread and executing a different thread.

14. (Currently amended) The method of claim 13, wherein in response to detecting that said given thread does not include said different identifier, continuing execution of said given thread.

15-16. (Cancelled)

17. (Currently amended) A method of generating low-level instructions executable by a processor, said method comprising:

providing a computer program including high-level programming instructions;
detecting whether an indicator is included within said computer program; and
in response to detecting said indicator, generating a low-level instruction having
[[an]] a specific identifier corresponding to said indicator, wherein said low-level identifier is configured to cause said processor to selectively continue execution of a given thread by:
suspending execution of said given thread and executing a different thread
in response to receiving a global execution parameter; and
overriding said global execution parameter and continuing execution of
said given thread in response to detecting said specific identifier.

18. (Original) The method of claim 17, wherein said indicator is a compiler directive.

19. (Original) The method of claim 17, wherein said indicator is an assembly language subroutine.

20. (Original) The method of claim 17, wherein said indicator is a unique high-level instruction.

21. (Currently amended) A machine-readable medium comprising program instructions, wherein said program instructions are executable by a processor to:

- execute one or more threads;
- detect whether a given thread includes [[an]] a specific identifier; [[and]]
- selectively continue execution of said given thread depending upon whether said identifier is detected;
- suspend execution of said given thread and executing a different thread in response to receiving a global execution parameter; and
- override said global execution parameter and continuing execution of said given thread in response to detecting said specific identifier.

22. (Currently amended) A processor comprising:

- means for executing one or more threads;
- means for detecting whether a given thread includes [[an]] a specific identifier; [[and]]
- means for selectively continuing execution of said given thread depending upon whether said specific identifier is detected;
- means for suspending execution of said given thread and executing a different thread in response to receiving a global execution parameter; and
- means for overriding said global execution parameter and continuing execution of said given thread in response to detecting said specific identifier.